A TCL/TK WIDGET FOR DISPLAY OF MEDM SCREENS*

R. Soliday, ANL, Argonne, IL 60439, U.S.A.

Abstract

A new Tcl/Tk widget has been created to display MEDM screens inside a Tcl/Tk application. Tcl/Tk parses the MEDM input files and the appropriate widgets are created and linked to the associated process variables. One advantage of this approach is that an X-Windows emulator is not required to view and manipulate the MEDM screen under a Windows operating system. Another benefit is that the MEDM screen can now be tightly integrated into a scripting language to attach higher-level logic to various process variable manipulations. Further details and examples of the new widget will be discussed.

INTRODUCTION

While preparing for a presentation at the Getting Started with EPICS (Experimental Physics and Industrial Control System) [1] lecture series at the Advanced Photon Source (APS) on the Operations and Analysis Group (OAG) Tcl/Tk interpreter, a comprehensive demo was created that would show how to use the channel-access (CA) extension in OAG Tcl/Tk. The demo was created to be visually impressive and showcase many of the features of the CA extension. The demo consisted of a new type of Tcl/Tk widget to display a Motif Editor and Display Manager (MEDM) screen [2] with native Tcl/Tk code. This was done because users at the APS were already familiar with MEDM and stark contrasts could be made about the ease of programming in Tcl/Tk compared to C, which is the language in which MEDM was written. For the purposes of this paper I will refer to the OAG Tcl/Tk MEDM screen as the MEDM widget while the original MEDM program will simply be referred to as MEDM.

Tcl/Tk is a highly used programming language for controlling machine operations at the APS. Tcl (Tool Command Language) is a scripting language that is human readable, scalable, fast, extensible, and cross platform. Tk is a graphical user interface (GUI) toolkit and is an extension of Tcl. Many other scripting languages use Tk as their GUI. Without the need for compilations between minor changes, the development of software is relatively fast. GUIs require just a handful of commands to define them because the Tk library provides a higher-level interface than most standard C library toolkits. The OAG versions of Tcl and Tcl/Tk are called oagtclsh and oagwish. These versions include EPICS CA and Self Describing Data Sets (SDDS) [3] extensions as well as others. The OAG Tcl/Tk interpreter also includes an extensive collection of general-purpose graphical and non-graphical Tcl/Tk procedures.

To display an MEDM widget inside an OAG Tcl/Tk application, the name of an MEDM ASCII Display List (ADL) file is passed to the widget, which in turn parses the file using native Tcl code. Then the appropriate child widgets are created to look and behave identically to how they would have had MEDM been used to display the screen instead of OAG Tcl/Tk. Not every last feature of MEDM has been mimicked, but most of it has been, and additions can be made relatively quickly. While this widget was not intended to replace well-tested code such as MEDM, there is one important scenario where this approach has a clear advantage. Unlike MEDM, no installations of the Exceed X-server and X libraries [4] are required to display the screens on Windows computers using OAG Tcl/Tk.

TCL CHANNEL ACCESS EXTENSION

Bob Daly wrote the original version of the Tcl CA extension. Claude Saunders rewrote it while he was still with the Operations and Analysis Group in 1996. Over the years this code has been expanded upon and improved. With it you can take a standard Tcl/Tk interpreter and dynamically load the extension to provide commands to access process variables (PVs). PVs can be linked to Tcl/Tk variables. The values can be changed locally and in turn set the corresponding PV in the IOC. PV values in an IOC can also be monitored so that the corresponding Tcl/Tk variables can track PV value changes.

One of the main advantages to this design is that no external programs are required in order to interact with PVs, which results in a noticeable increase in speed. There is also a benefit to be gained because only one CA search per PV is required for the duration of the execution of the Tcl/Tk script. If an external program were used, it would have to reconnect to PVs each time it was executed. So by using the CA extension, the network load can be reduced.

The Tcl/Tk CA extension is written in C to gain a speed advantage compared to straight Tcl/Tk scripting code. This also allows it to interface directly with the EPICS code. The CA extension has commands to search, poll, and set PVs. These commands come in two flavors, blocking and nonblocking. The blocking version blocks execution until all the PVs have responded. These commands are more straightforward and therefore easier to use. The nonblocking commands keep the GUI responsive if there are any timeout issues where a PV is not responding. They also make it possible to isolate a non-responsive PV from a larger list of PVs. In the case of the MEDM widget, nonblocking commands were used because GUI responsiveness and identification of non-responsive PVs was important. Another type of command that the CA extension provides is the PV monitoring commands, which also come in two flavors. The first will update the associated Tcl variable when it is read. This

*Work is supported by U.S. Department of Energy, Office of Basic Energy Sciences, under Contract No. W-31-109-ENG-38.

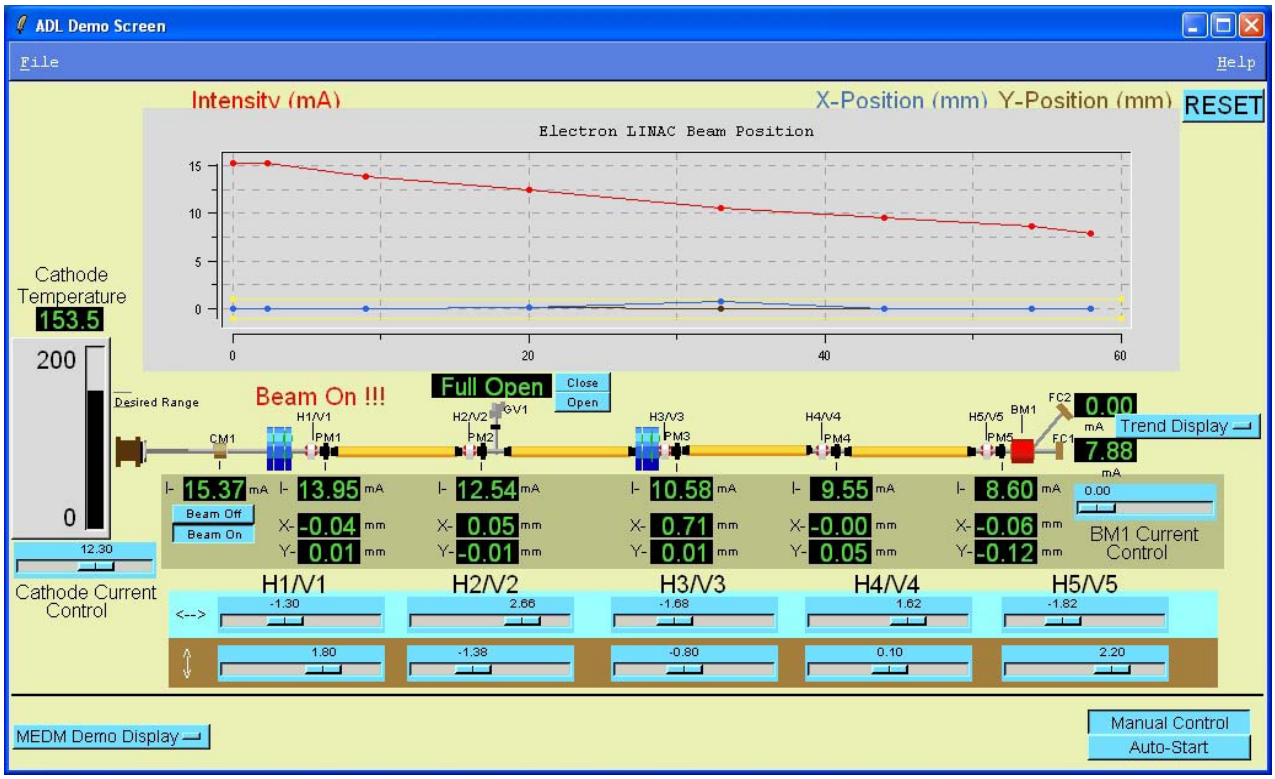

Figure 1: OAG Tcl/Tk application with MEDM widget.

means that if the variable is displayed in the GUI, it will not automatically change when the PV changes. In the MEDM widget this command is used in conjunction with a procedure that runs every half-second to update the widgets. This method is used to reduce screen flicker in child widgets that may otherwise update at an unnecessarily fast rate. The second type of monitoring command updates the Tcl variable and any widgets with which the variable is associated. These are used in widgets that are normally only going to be changed by the user, such as entry boxes.

MEDM WIDGET

To create an MEDM widget inside of an OAG Tcl/Tk application such as that shown in Figure 1, an MEDM ADL file is required. The command to create the widget is called **APSCreateADLScreen** and accepts three arguments. The first is the name of the MEDM ADL file, the second is the name of the parent OAG Tcl/Tk widget, and the third is an optional macro command. The macro option performs the same function as it does in the MEDM code. It replaces one string with another in the ADL file before it is parsed.

The main backdrop of the MEDM widget is a Tcl/Tk canvas widget on which all other widgets are drawn. After creating the canvas widget, the procedure **APSParseADLFile** is called. This is passed the name of the canvas, the macro definition, and a unique ID so that multiple MEDM widgets can be created in the same application. This procedure calls various other procedures to parse and extract all the required information from the ADL file. The various child widgets are then created on the canvas, such as shapes, plots, entry boxes, labels, buttons, etc. as the ADL file instructs. This also includes setting the proper geometry and color map as defined in the ADL file. The child widgets are also associated with Tcl/Tk variables that will soon be connected to PVs. Most of the child widgets are created using built-in Tcl/Tk widgets and then changing some default properties to make them appear like standard MEDM widgets. Two exceptions to this are the plot widget and the bar widget. These two types of widgets were designed from scratch to look and behave like the standard MEDM widgets.

The next step in **APSCreateADLScreen** is to create three parallel lists of Tcl/Tk variables, PV names, and PV connection states. It then runs the CA extension command to link to all the PV variables in a nonblocking manner. At this point it draws the entire MEDM widget on the screen. Prior to this the widget was hidden until it could be displayed in its entirety when it was finished. A timed-event procedure is then initiated to check which PVs have connected. When they have connected, the connection state variable is updated appropriately. Also the precision of the variable is detected and used in the display of the value. The maximum and minimum allowed values are also used now to set the proper limits to bars and scales. With radio buttons the choices are read and the appropriate radio buttons are created. Once any

PV has connected, the appropriate CA monitoring command is initiated on that PV. If one or more PVs fail to connect, the timed event procedure to check for connections is setup to run again after 0.1 seconds.

A separate timed-event procedure is also set up to run at half-second intervals. Any changes to PVs associated with labels, bars, and plots will now be reflected in the widget when this procedure runs. When the user interacts with child widgets that can be changed, such as entry boxes, scales, and buttons, the CA extension command to set the PV is called to update the PV to the desired value.

COMPARISON

The behavior of the OAG Tcl/Tk MEDM widget for most ADL files is almost the same as the corresponding file being displayed by MEDM. A few differences still remain, e.g., support for background images has not yet been included. Also there is no editing mode for the MEDM widget.

The GUI appearance is extremely similar. Much of this is due to the font selection algorithm that was tested by a trial and error method to reproduce the same font size and style. Different child widgets have different algorithms, but they are all based on the height of the child widget.

The number of lines of code for the OAG Tcl/Tk MEDM widget is 1865 while MEDM has 67,195 lines of code. This is a great example of Tcl/Tk being easier and quicker to code than C code. In fact, the entire project took just over a week to complete. To be fair, the code to allow editing in MEDM is extensive.

While the MEDM does have an execution speed advantage, the OAG Tcl/Tk MEDM widget is not slow and does not occupy too much CPU time. The casual user should not notice a difference in GUI responsiveness.

MEDM was designed for UNIX systems but will run on Windows computers if the Exceed X-server is already installed. The OAG Tcl/Tk MEDM can run on UNIX systems as well as Windows computers without the Exceed X-server. This can potentially lead to a cost of deployment savings for users and their employers since the Exceed X-server is not available for free.

REFERENCES

[1] L. Dalesio, M. Kramer, A. Kozubal, “EPICS Architecture,” Proc. ICALEPCS 1991 Conference, Tsukuba, Japan, pp. 278-281 (1992).

[2] K. Evans, Jr., “An Overview of MEDM,” Proc. ICALEPCS 1999 Conference, Trieste, Italy, pp. 466-468 (1999), www.jacow.org.

[3] M. Borland and L. Emery, “The Self-Describing Data Sets File Protocol and Program Toolkit,” Proc ICALEPCS 1995 Conference, Chicago, Illinois, pp. 653-662 (1996).

[4] Exceed is a product of Hummingbird Ltd., Toronto, Ontario, Canada.

